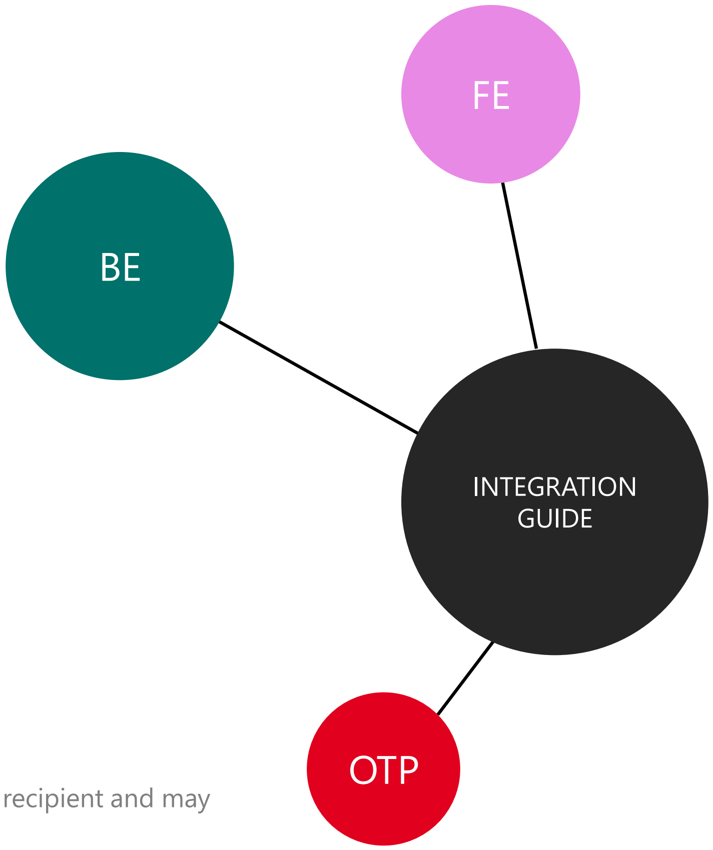
Wibiocard web authentication integration guide

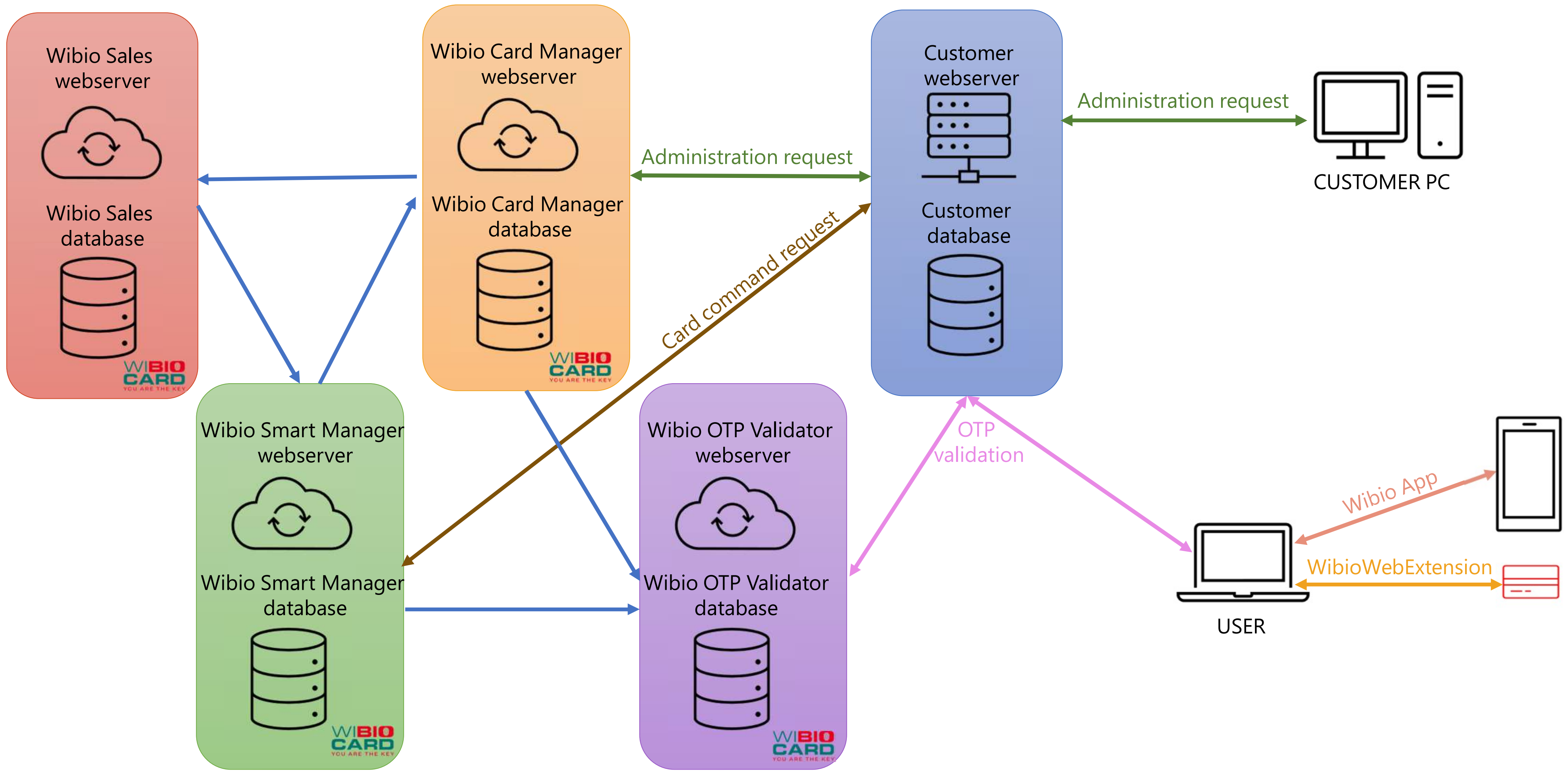


V. 20241113A

General workflow

Wibio Sales (website)	**Wibio Card manager (website) or Customer website**	**Wibio Smart manager (backend server)**	**Wibio Otp validator (backend server)**
New order →	→	Load CARDS ID	
	New company owner →	→	Create the Realm
	New user →	Manage user info →	Create user
	Card activation →	Generate token data →	Save token
		↓	
	Decrypt on native element and run apdu commands ⇠	Generate encrypted personalization APDU commands	
	Assign user/token →	→	Token activation
	New OTP web access request →	→	Otp validation

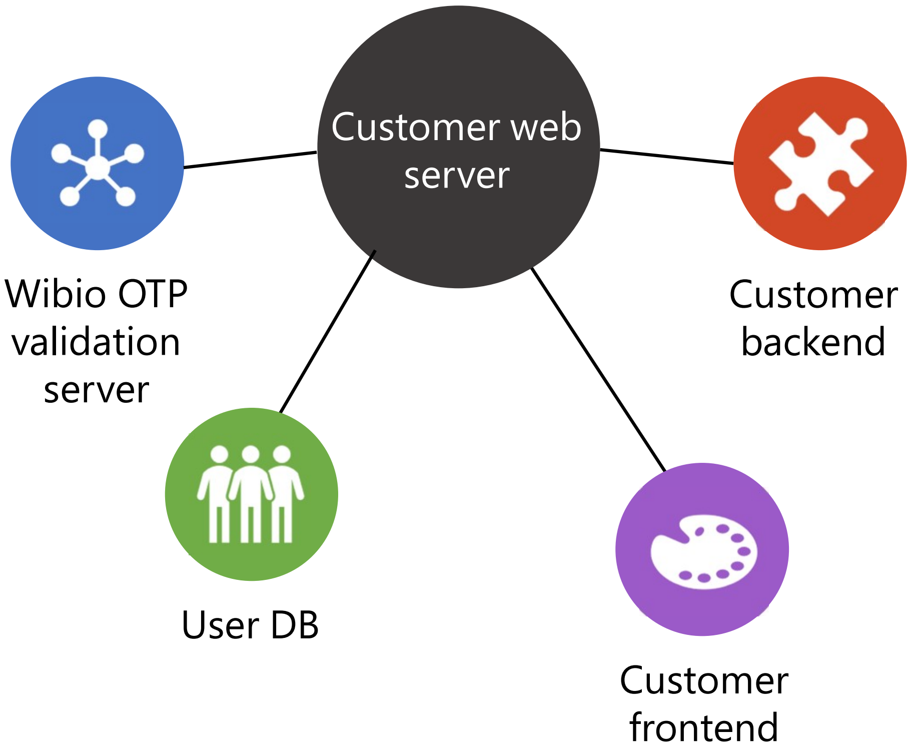
General Infrastructure



TO DO LIST

Integration with the personal website takes place on several fronts

- ❑ The frontend, where it is necessary to integrate the communication functions with the smartcard using the Wibiocard web extension
- ❑ The backend, where it is necessary to integrate the communication functions with the Wibiocard OTP validation server in order to configure, activate and manage the tokens
- ❑ The database, where you need to add a card-to-user association value into your DB



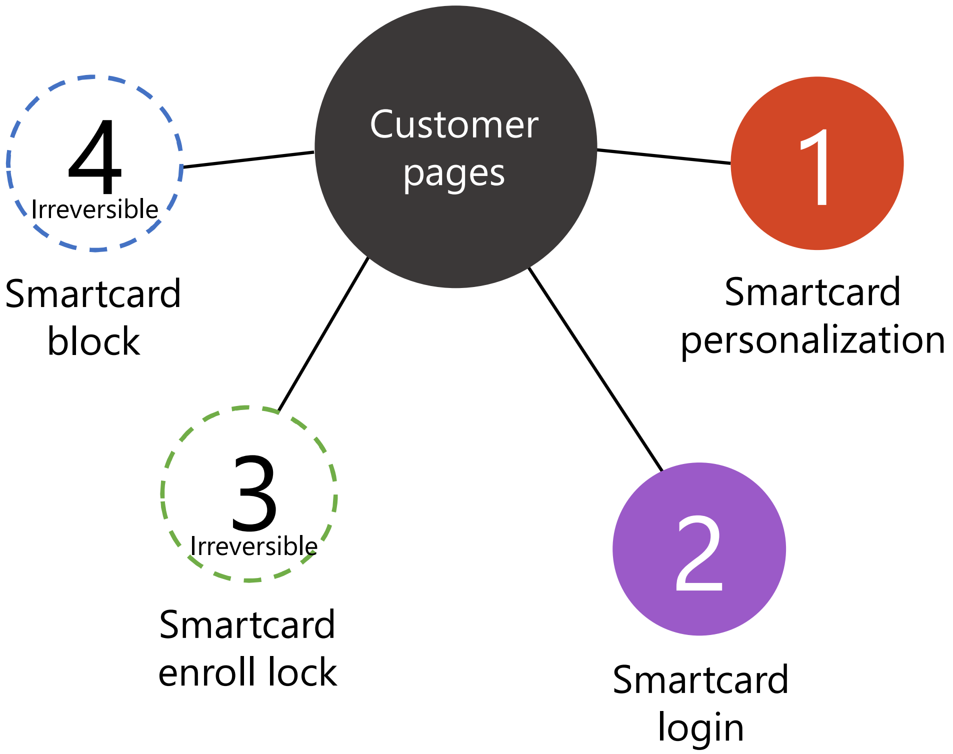
Frontend integration

1. The cloud process of **customizing and personalization** of a smartcard is used to make the smartcard active and to record the token data on the OTP validation server

2. The cloud **login** process obtains an OTP code from the smartcard which, via API call from the webserver, is used to validate the generated code.

3. The cloud process of **locking the enroll functionality** allows you to make the smartcard inviolable. After this operation, it will no longer be possible to change the fingerprints registered on the card in any way.

4. The cloud process of **blocking** the smartcard prevents any further use of the smartcard. This action is not on card, but on OTP validation server



FRONT END - customizing and personalization

When the smartcard arrives at the customer, it is completely empty, with no usable tokens until it is initialized.

Initialization takes place through a secure process that uses the native component of the web extension to decrypt the smartcard commands (APDU) that are issued by the server.

The Javascript component of the web page must then request specific sequences of commands via API call to the “WibiocardSmartManagerServer” server, which responds with the encrypted command, it is then sent to the native component that decrypts it, sends it to the smartcard and parses the result obtained.

During the smartcard customization phase where OTP access tokens are generated, keys are simultaneously sent to the “OtpValidationServer” server and the frontend using 2 different encryption algorithms.

After electrical personalization, the smartcard can be used. The fingerprint registration of the cardholder can be performed before or after this operation.

https://github.com/Wibiocard/WibioWebExtension_integration_example

TYPE F CARD COMMANDS:

First read the card type and get/set the card ID
[SelectWibioApp] [GetCardUuid][SetCardUid {uuid="+cardId+"}]

Then need to be ensure that the card is empty
[SelectBeCard][ReadSequenceInfo]

Finally personalize the card with the CardID printed on it
[SelectBeCard][PersonalizeF {token="+cardId+"}][ReadSequenceInfo]

FRONT END – protecting from froud

When the card arrives at the customer, it is equipped with an applet that allows fingerprints to be registered using the special enroll tool.

After the card owner has done this operation, the applet can be removed so that no other fingerprint editing operations can be performed.

Be careful, this operation is not reversible in any way.

OPTIONAL

https://github.com/Wibiocard/WibioWebExtension_integration_example

TYPE F CARD COMMANDS:

First test the correct fingerprint validation by selecting the sequence to validate
[SelectBeCard] [ReadSequenceInfo]

Then try to get an otp with each finger on the given sequences
[SelectBeCard] [ReadOtpToken {OtpMode="+sequence+"}][ReadOtpToken {OtpMode="+sequence+"}]

Finally remove the applet
[ProtectCard]

FRONT END – get the OTP from card

When the customization process is complete and the user has enrolled the fingerprints, an OTP request can be made on card then the authentication server **must validate** them.

On smartcard it is possible to set more than one OTP sequence, so before executing the final request it is necessary to obtain the list of sequences and select one that will be used for the generation of the OTP. If the customer choose to not have more than one OTP sequence on the same card it can store this information on database and set this data directly into the javascript page.

https://github.com/Wibiocard/WibioWebExtension_integration_example

TYPE F CARD COMMANDS:

First get the sequences

```
[SelectBeCard] [ReadSequenceInfo]
```

Finally get the OTP fir a choosen sequence

```
[SelectBeCard] [ReadOtpToken {OtpMode="+sequence+"}]
```

BACK END – work with the OTP validator server

In the backend, you need to create the endpoints that will communicate with the validator server.

These features can then be invoked from the frontend to perform user and token management operations on the validator server.

All management functions can only be performed after authentication with admin user and password. Authentication returns a csrf_access_token that must be used as the data header in all subsequent calls.

```
curl_setopt($ch, CURLOPT_HTTPHEADER, array('X-CSRF-TOKEN: ' . $csrf_access_token));
```

https://github.com/Wibiocard/WibioValidationServer_integration_sample

PHP EXAMPLE:

```
$WibioOtpRequestHelper = new WibioOtpRequestHelper();
$WibioOtpRequestHelper->init();
$resp = $WibioOtpRequestHelper->adminShowAll();
$WibioOtpRequestHelper = null;
```

BACK END – available functions

class WibioOtpRequestHelper		
__construct	__destruct	init
adminAssign	adminCheck	adminDisableUser
adminDisableToken	adminEnableUser	adminEnableToken
adminGetTokenOwner	adminRemoveUser	adminRemoveToken
adminResetUser	adminResetToken	adminResyncUser
adminResyncToken	adminSetUser	adminSetToken
adminShowAll	adminShowUser	adminShowToken
adminTokenRealm	adminUnassignToken	adminUserList

https://github.com/Wibiocard/WibioValidationServer_integration_sample

Follow the gitHub link for more informations and for functions details

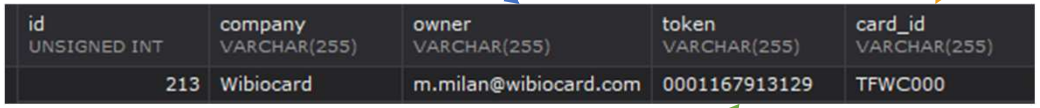
DATABASE

In your database you need to add only one field: the card Uuid that must be associated to the smartcard owner user.

The sequence ID can be saved optionally;
if you do not want to allow user to choose the otp sequence on the card to generate the OTP.
Instead, you can retrieve the sequence ID each time from the smartcard

Email is used as unique id for the user

CardId is printed on card and is the unique card identification



id UNSIGNED INT	company VARCHAR(255)	owner VARCHAR(255)	token VARCHAR(255)	card_id VARCHAR(255)
213	Wibiocard	m.milan@wibiocard.com	0001167913129	TFWC000

A card can have more then one tokens that can be used to generate OTP for different services, if you want you can save it on your database

OTP VALIDATION

To validate an OTP you can simply generate an API GET request to the validator server.

Do not perform this API request from the frontend! It work but it is unsecure…

https://otpsandbox.wibiocard.com/validate/check?user=___&pass=___&realm=___

The parameters required are:

- `user – (required) the username`
- `pass – (required) the OTP`
- `realm – (required) the realm`

The **realm** is the given name of our company, if do not have this information please contact Wibiocard.

GOOD WORK